

CLAIM AMENDMENTS

Claim Amendment Summary

Claims pending

- Before this Amendment: Claims 1-42 and 45-46.
- After this Amendment: Claims 1-28, 34-42, and 45-46.

Non-Elected, Canceled, or Withdrawn claims: 29-33.

Amended claims: 1, 13, 34, 40, and 45.

New claims: none.

Claims:

1. (CURRENTLY AMENDED) A kernel emulator for non-native program modules, the kernel emulator being software comprised of computer-executable instructions that are tangibly embodied on one or more computer-readable media and the kernel emulator comprising:

an interceptor configured to intercept kernel calls from non-native program modules;

a call-converter configured to convert non-native kernel calls intercepted by the interceptor into native kernel calls.

2. (ORIGINAL) An emulator as recited in claim 1, wherein the call-converter comprises a translator configured to translate a non-native paradigm for passing parameters into a native paradigm for passing parameters.

3. (ORIGINAL) An emulator as recited in claim 1, wherein the call-converter comprises a translator configured to translate non-native CPU instructions into native CPU instructions.

4. (ORIGINAL) An emulator as recited in claim 1, wherein the call-converter comprises a translator configured to translate addresses from non-native length into native length.

5. (ORIGINAL) An emulator as recited in claim 1, wherein the call-converter comprises an argument-converter configured to convert non-native argument format into native argument format.

6. (ORIGINAL) An emulator as recited in claim 1, wherein the call-converter comprises a translator configured to translate words from non-native word size into native word size.

7. (ORIGINAL) An emulator as recited in claim 1 further comprising a memory constrainer configured to limit addressable memory to a range addressable by non-native program modules.

1 **8. (ORIGINAL)** An emulator as recited in claim 1 further
2 comprising a shared-memory manager configured to manage memory space that is
3 accessible to both native and non-native program modules.
4

5 **9. (ORIGINAL)** An emulator as recited in claim 1 further
6 comprising a shared-memory manager configured to synchronize a native shared
7 data structure with a non-native shared data structure.
8

9 **10. (PREVIOUSLY PRESENTED)** An emulator as recited in
10 claim 1 further comprising a shared-memory manager configured to manage
11 memory space that is accessible to both native and non-native program modules,
12 wherein the shared-memory manager maps versions of process shared data
13 structures (process SDSs) and versions of thread shared data structures (thread
14 SDSs) between native and non-native program modules.
15

16 **11. (ORIGINAL)** An operating system on a computer-readable
17 medium, comprising:

18 a native kernel configured to receive calls from native program modules;

19 a kernel emulator as recited in claim 1 configured to receive calls from non-
20 native program modules.
21
22
23
24
25

1
2 **12. (ORIGINAL)** An operating system on a computer-readable
3 medium, comprising:

4 a native kernel configured to receive calls from native APIs;

5 a kernel emulator as recited in claim 1 configured to receive calls from non-
6 native APIs.

7
8 **13. (CURRENTLY AMENDED)** A method of emulating a kernel
9 for non-native program modules, the method comprising:

10 intercepting kernel calls from non-native program modules, the kernel calls
11 calling a kernel being software comprised of computer-executable instructions that
12 are tangibly embodied on one or more computer-readable media;

13 converting the intercepted non-native kernel calls into native kernel calls.

14
15 **14. (ORIGINAL)** A method as recited in claim 13, wherein the
16 converting step comprises translating a non-native paradigm for passing
17 parameters into a native paradigm for passing parameters.

18
19 **15. (ORIGINAL)** A method as recited in claim 13, wherein the
20 converting step comprises translating non-native CPU instructions into native
21 CPU instructions.

1 **16. (ORIGINAL)** A method as recited in claim 13, wherein the
2 converting step comprises translating addresses from non-native length into native
3 length.
4

5 **17. (ORIGINAL)** A method as recited in claim 13, wherein the
6 converting step comprises translating words from non-native word size into native
7 word size.
8

9 **18. (ORIGINAL)** A method as recited in claim 13 further
10 comprising limiting addressable memory to a range addressable by non-native
11 program modules.
12

13 **19. (ORIGINAL)** A method as recited in claim 13 further
14 comprising synchronizing a native shared data structure with a non-native shared
15 data structure.
16

17 **20. (ORIGINAL)** A method as recited in claim 13 further
18 comprising mapping versions of process shared data structures (SDSs) between
19 native and non-native program modules.
20

21 **21. (ORIGINAL)** A method as recited in claim 20, wherein a
22 process SDS of a native program module includes a pointer to a process SDS of a
23 non-native program module.
24
25

1 **22. (ORIGINAL)** A method as recited in claim 20, wherein a
2 process SDS of a non-native program module includes a pointer to a process SDS
3 of a native program module.
4

5 **23. (ORIGINAL)** A method as recited in claim 13 further
6 comprising mapping versions of thread shared data structures (SDSs) data
7 structure between native and non-native program modules.
8

9 **24. (ORIGINAL)** A method as recited in claim 23, wherein a
10 thread SDS of a native program module includes a pointer to a thread SDS of a
11 non-native program module.
12

13 **25. (ORIGINAL)** A method as recited in claim 23, wherein a
14 thread SDS of a non-native program module includes a pointer to a thread SDS of
15 a native program module.
16

17 **26. (ORIGINAL)** A computer comprising one or more computer-
18 readable media having computer-executable instructions that, when executed by
19 the computer, perform the method as recited in claim 13.
20

21 **27. (ORIGINAL)** A computer-readable medium having computer-
22 executable instructions that, when executed by a computer, performs the method
23 as recited in claim 13.
24
25

1 **28. (ORIGINAL)** An operating system embodied on a computer-
2 readable medium having computer-executable instructions that, when executed by
3 a computer, performs the method as recited in claim 13.

4
5 **29. (CANCELED)** A method comprising:
6 determining whether an initiating program module is a native or
7 non-native;
8 if the initiating program is non-native:
9 limiting available memory to a range that is addressable by
10 the non-native program module, that range of addressable memory
11 being less than the available memory;
12 establishing non-native a version of a shared memory data
13 structure that may be synchronized with a native version of the same
14 shared memory data structure.

15
16 **30. (CANCELED)** A method as recited in claim 29 further
17 comprising:
18 intercepting kernel calls from the non-native program module;
19 converting the intercepted non-native kernel calls into native kernel calls.

20
21 **31. (CANCELED)** A method as recited in claim 29 further
22 comprising emulating a non-native kernel for which kernel calls from the non-
23 native program module are intended.
24
25

32. (CANCELED) A computer comprising one or more computer-readable media having computer-executable instructions that, when executed by the computer, perform the method as recited in claim 29.

33. (CANCELED) A computer-readable medium having computer-executable instructions that, when executed by a computer, performs the method as recited in claim 29.

34. (CURRENTLY AMENDED) A method comprising emulating a non-native kernel for a native computing platform so that kernel calls from non-native applications are translated into calls to a native kernel, the native kernel being software comprised of computer-executable instructions that are tangibly embodied on one or more computer-readable media.

35. (ORIGINAL) A method as recited in claim 34, wherein the emulating step comprises:

translating non-native CPU instructions into native CPU instructions;

translating addresses from non-native length into native length;

limiting addressable memory to a range addressable by non-native program modules.

1 **36. (ORIGINAL)** A method as recited in claim 35, wherein the
2 emulating step further comprises translating a non-native paradigm for passing
3 parameters into a native paradigm for passing parameters.

4
5 **37. (ORIGINAL)** A method as recited in claim 34, wherein the
6 converting step further comprises translating words from non-native word size into
7 native word size.

8
9 **38. (ORIGINAL)** A computer comprising one or more computer-
10 readable media having computer-executable instructions that, when executed by
11 the computer, perform the method as recited in claim 34.

12
13 **39. (ORIGINAL)** A computer-readable medium having computer-
14 executable instructions that, when executed by a computer, performs the method
15 as recited in claim 34.

16
17 **40. (CURRENTLY AMENDED)** A kernel emulator configured to
18 emulate a non-native kernel for a native computing platform so that kernel calls
19 from non-native applications are translated into calls to a native kernel, the kernel
20 emulator being software comprised of computer-executable instructions that are
21 tangibly embodied on one or more computer-readable media.
22
23
24
25

1 **41. (ORIGINAL)** An emulator as recited in claim 40, wherein the
2 emulator comprises:

3 an instruction-translator configured to translate non-native CPU
4 instructions into native CPU instructions;

5 an address-translator configured to translate addresses from non-native
6 length into native length;

7 an memory constrainer configured to limit addressable memory to a range
8 addressable by non-native program modules.

9
10 **42. (PREVIOUSLY PRESENTED)** An operating system on a
11 computer-readable medium, comprising:

12 a native kernel configured to receive calls from native program modules;

13 a kernel emulator as recited in claim 40 configured to receive calls from
14 non-native program modules.

15
16 **43. (CANCELED)**

17
18
19 **44. (CANCELED)**

1 **45. (CURRENTLY AMENDED)** A kernel emulator for non-native
2 program modules, the kernel emulator being software comprised of computer-
3 executable instructions that are tangibly embodied on one or more computer-
4 readable media and the kernel emulator comprising:

5 an interceptor configured to intercept kernel calls from non-native program
6 modules;

7 a call-converter configured to convert non-native kernel calls intercepted by
8 the interceptor into native kernel calls, wherein the call-converter comprises:

9 an instruction-translator configured to translate non-native CPU
10 instructions into native CPU instructions;

11 an address-translator configured to translate addresses from non-
12 native length into native length.

13
14 **46. (ORIGINAL)** An operating system on a computer-readable
15 medium, comprising:

16 a native kernel configured to receive calls from native program modules;

17 a kernel emulator as recited in claim 45 configured to receive calls from
18 non-native program modules.
19